Detection of LSB Steganography via File Serialization in Powershell

Joe Petroske
BSides MSP
June 11, 2016

BLUEPRINT

- $whoami
- $whatamitalkingabout
- How it actually works
- Steganalysis algorithm
- Creation of a Stegchannel
- Detection code and demo

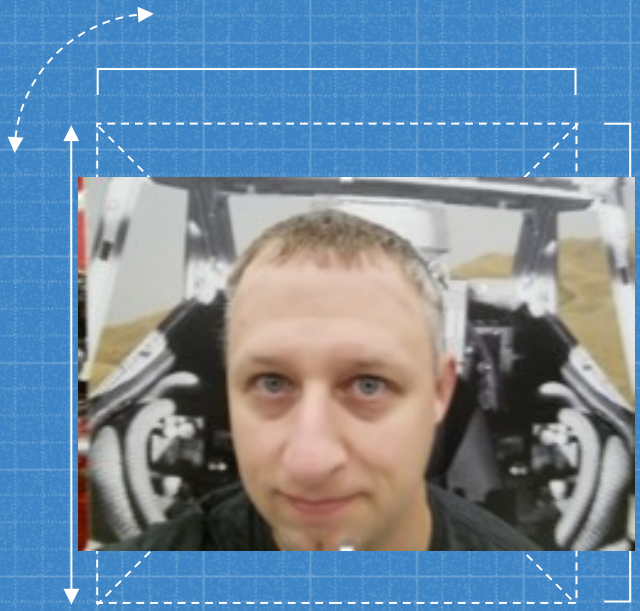
THANKS TO THE BSIDES MSP CREW

$whoami

Joe Petroske, <insert obligatory certification initials here>

Incident Handler, Target Cyber Fusion Center

<LinkedIn>



WHAT THIS IS *NOT*

- The only way to do this (multiple stego techniques exist)
- Novel (based on 16-year old research)
- Required (commercial tools exist)
- A tool recommendation
- A coding workshop
- Intended to be a language holy war

1

What The <expletive> Are You Talking About?

STEGANOGRAPHY DEFINED

The art and science of concealing a hidden message

- **Greek:**
  - ***steganos*** **= covered/hidden**
  - ***graphein*** **= writing**
  - **Same root as Stegosaurus "hidden lizard"**
  - **"Steganalysis" = the art and science of detecting hidden messages**
  - **"Stegasauranalysis" = the art and science of finding Stegosaruses (stegasauri?)**

"Now this end is called the thagomizer . . . after the late Thag Simmons."

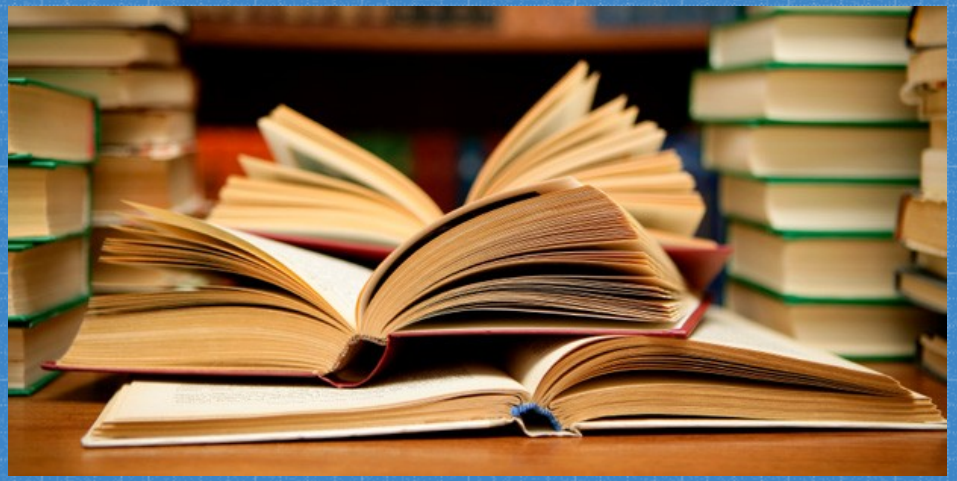
In this case, we will be discussing the technique of hiding an image file within another image file

TERMINOLOGY

- *Payload* = “the file we want to hide”
- *Carrier* = “the file that the payload is hidden in”
- *Channel/Stegchannel/Stegfile/ Package* =
- “the carrier with the payload hidden in it”
- *Encoding Density* = ratio of bits modified to total bits of carrier

WHY IS THIS RELEVANT?

- In this case... It’s an interesting academic topic



WHY IS THIS RELEVANT?

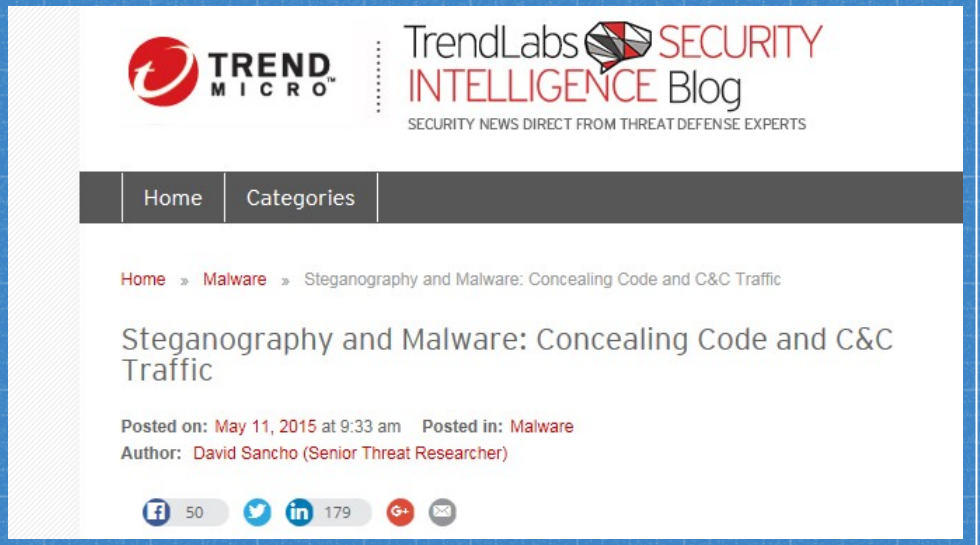
- But more sinister applications for steganography exist…
- …that military, intelligence, law enforcement, and security practictioners would care about…

WHY IS THIS RELEVANT?

- But more sinister applications for steganography exist…
- …that military, intelligence, law enforcement, and security practictioners would care about…
- Terrorism

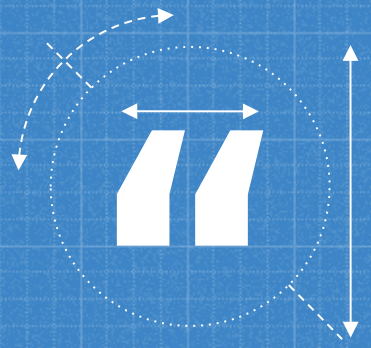
WHY IS THIS RELEVANT?

- But more sinister applications for steganography exist…
- …that military, intelligence, law enforcement, and security practictioners would care about…
- Malware delivery and control



WHY IS THIS RELEVANT?

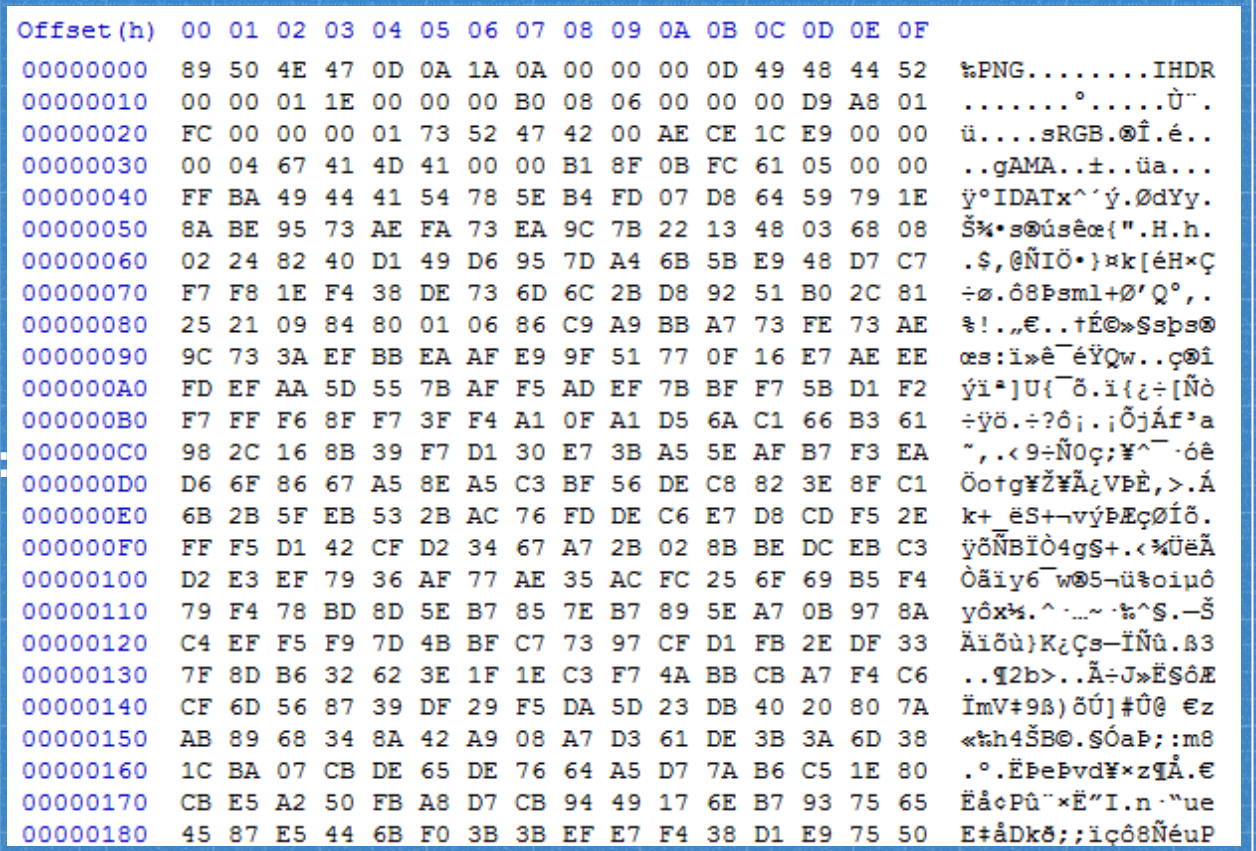
- But more sinister applications for steganography exist…
- …that military, intelligence, law enforcement, and security practictioners would care about…
- …and truly awful things



- SO LET’S GO DETECT SOME STEGANOGRAPHY

PRELUDE: WHAT ACTUALLY *IS* AN IMAGE FILE?

- Sure, it's a picture…
- But it's really a stream of bytes
- Crack one open in a hex editor:
  - Header
  - Data bits
  - Trailer



Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	
00000000	89	50	4E	47	0D	0A	1A	0A	00	00	00	0D	49	48	44	52	‰PNG.....IHDR
00000010	00	00	01	1E	00	00	00	B0	08	06	00	00	00	D9	A8	01	.....°.....Ù¨.
00000020	FC	00	00	00	01	73	52	47	42	00	AE	CE	1C	E9	00	00	ü....sRGB.®Î.é..
00000030	00	04	67	41	4D	41	00	00	B1	8F	0B	FC	61	05	00	00	..gAMA..±..üa...
00000040	FF	BA	49	44	41	54	78	5E	B4	FD	07	D8	64	59	79	1E	ÿºIDATx^´ý.ØdYy.
00000050	8A	BE	95	73	AE	FA	73	EA	9C	7B	22	13	48	03	68	08	Š¾•s®úsêœ{".H.h.
00000060	02	24	82	40	D1	49	D6	95	7D	A4	6B	5B	E9	48	D7	C7	.$‚@ÑIÖ•}¤k[éH×Ç
00000070	F7	F8	1E	F4	38	DE	73	6D	6C	2B	D8	92	51	B0	2C	81	÷ø.ô8Þsml+Ø'Q°,.
00000080	25	21	09	84	80	01	06	86	C9	A9	BB	A7	73	FE	73	AE	%!.„€..†É©»§sþs®
00000090	9C	73	3A	EF	BB	EA	AF	E9	9F	51	77	0F	16	E7	AE	EE	œs:ï»ê¯éŸQw..ç®î
000000A0	FD	EF	AA	5D	55	7B	AF	F5	AD	EF	7B	BF	F7	5B	D1	F2	ýiª]U{¯õ.ï{¿÷[Ñò
000000B0	F7	FF	F6	8F	F7	3F	F4	A1	0F	A1	D5	6A	C1	66	B3	61	÷ÿö.÷?ô¡.¡ÕjÁf³a
000000C0	98	2C	16	8B	39	F7	D1	30	E7	3B	A5	5E	AF	B7	F3	EA	˜,.‹9÷Ñ0ç;¥^¯·óê
000000D0	D6	6F	86	67	A5	8E	A5	C3	BF	56	DE	C8	82	3E	8F	C1	Öo†g¥Ž¥Ã¿VÞÈ‚>.Á
000000E0	6B	2B	5F	EB	53	2B	AC	76	FD	DE	C6	E7	D8	CD	F5	2E	k+_ëS+¬výÞÆçØÍõ.
000000F0	FF	F5	D1	42	CF	D2	34	67	A7	2B	02	8B	BE	DC	EB	C3	ÿõÑBÏÒ4g§+.‹¾ÜëÃ
00000100	D2	E3	EF	79	36	AF	77	AE	35	AC	FC	25	6F	69	B5	F4	Òãïy6¯w®5¬ü%oiµô
00000110	79	F4	78	BD	8D	5E	B7	85	7E	B7	89	5E	A7	0B	97	8A	yôx½.^·…~·‰^§.—Š
00000120	C4	EF	F5	F9	7D	4B	BF	C7	73	97	CF	D1	FB	2E	DF	33	Äiõù}K¿Çs—ÏÑû.ß3
00000130	7F	8D	B6	32	62	3E	1F	1E	C3	F7	4A	BB	CB	A7	F4	C6	..¶2b>..Ã÷J»Ë§ôÆ
00000140	CF	6D	56	87	39	DF	29	F5	DA	5D	23	DB	40	20	80	7A	ÏmV‡9ß)õÚ]#Û@ €z
00000150	AB	89	68	34	8A	42	A9	08	A7	D3	61	DE	3B	3A	6D	38	«‰h4ŠB©.§ÓaÞ;:m8
00000160	1C	BA	07	CB	DE	65	DE	76	64	A5	D7	7A	B6	C5	1E	80	.º.ËÞeÞvd¥×z¶Å.€
00000170	CB	E5	A2	50	FB	A8	D7	CB	94	49	17	6E	B7	93	75	65	Ëå¢Pû¨×Ë”I.n·“ue
00000180	45	87	E5	44	6B	F0	3B	3B	EF	E7	F4	38	D1	E9	75	50	E‡åDkð;;ïçô8ÑéuP

COOL. SO WHAT'S IN THE DATA BITS?

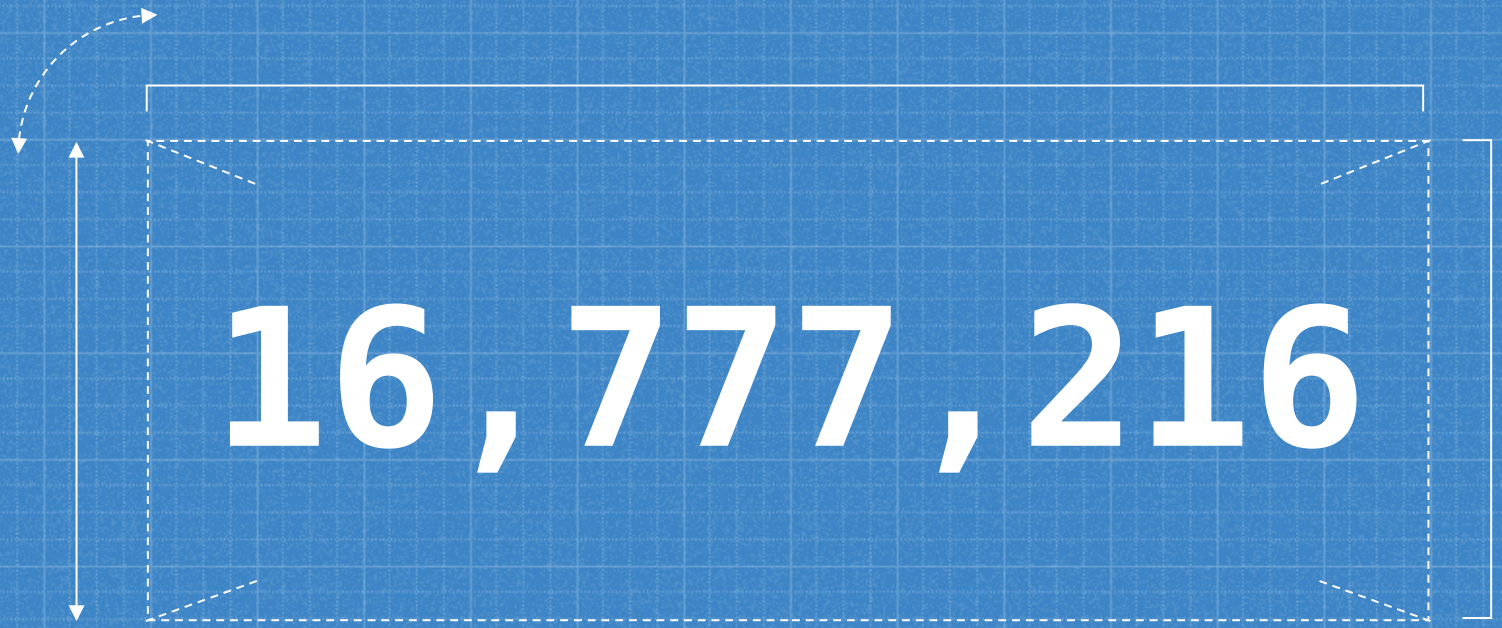
- Modern true-color images are 24-bit color
  - That's 3 bytes: 1 for Red, 1 for Green, 1 for Blue - RGB
  - (Then maybe one more byte for alpha – RGBA)
  - .bmp format is literally a map of bits
  - Hex string 78 9C AC BC 69 90 65 D9 71 (3 pixels)

RED	GREEN	BLUE
78	9C	AC
BC	69	90
65	D9	71

- Bytewise, that's
- 789CAC

- 65D971

- BC6990



Whoa! That’s a big number!

HOW DO YOU HIDE A FILE WITHIN A FILE?

- 24-bit color = 16,777,216 unique colors
- Yeah, but... Humans can only see about 10,000,000 colors
- I can personally see only 2^4 colors
- Changing the Least Significant Bit (LSB) of each color channel is undetectable to human eyes
  - This is “FF4411”

  - This is “FE4310”

- This means that we can hide 3 bits of data in each pixel (out of possible 24)

LSB STEGANOGRAPHY

- LSB image steganography is done by serializing the file you wish to hide, and replacing the LSB of each carrier pixel with the bits of your serialized file
- If done correctly, the stegfile is indistinguishable from the original carrier image!
- AND NOW WE HAVE A HIDDEN IMAGE THAT CANNOT BE DETECTED WITH THE NAKED EYE

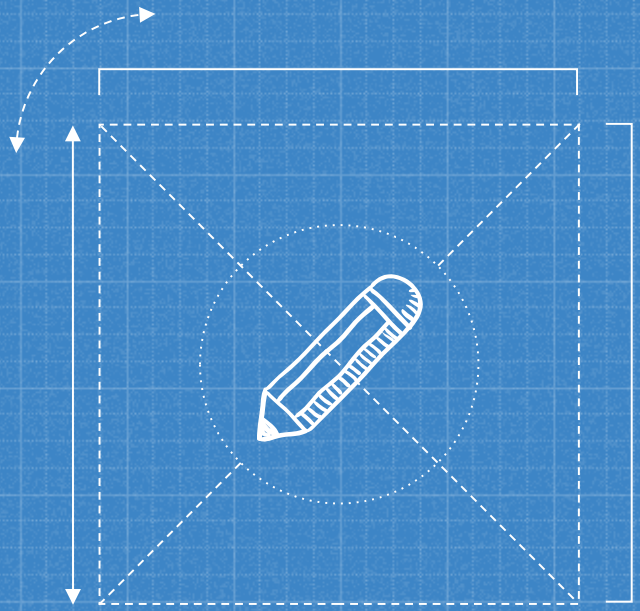
2 How It Actually Works

EFFECTIVE COLOR PALETTE

- There's 16777216 (2^{24}) potential unique colors in a true-color image
  - In practice, the actual palette size is FAR smaller than that
- Other than artifact borders, 2 adjacent pixels are often identical

WHY LSB STEGO WORKS

- If colors on an image were randomized, their LSBs would be too
  - We just showed that is not the case
  - Many adjacent pixels are either identical, or WAY more different than 1 bit per channel
- If you consider the payload image to be effectively random noise…
- This means the LSBs are as well!



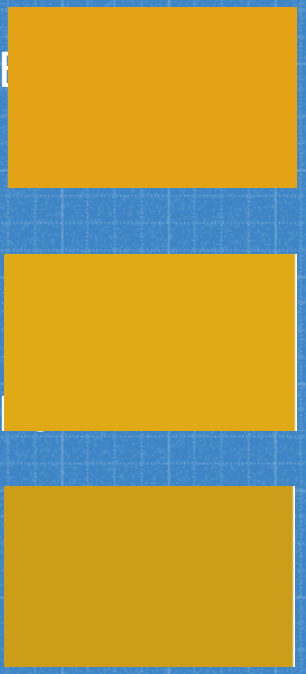
BIG CONCEPT, YO

An artifact of LSB encoding is the creation of multiple “close pairs” of colored pixels

REMEMBER THIS DOGE

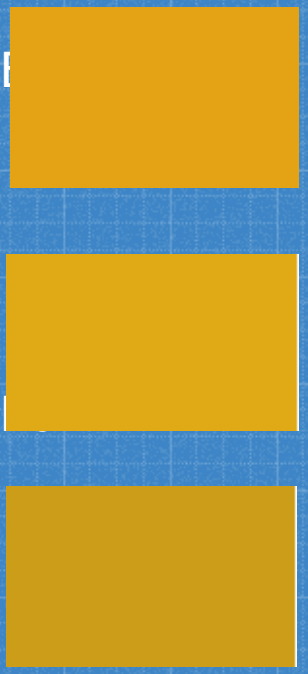
CREATION OF A CLOSE COLOR PAIR

- The 24-bit .bmp format actually stores pixels in B, G, R order
- Consider the following bitmap stream of 3 consecutive pixels:
- 15 A3 E4 16 AA E0 19 9D CC
  - 15 A3 E
  - 16 AA E
  - 19 9D CC



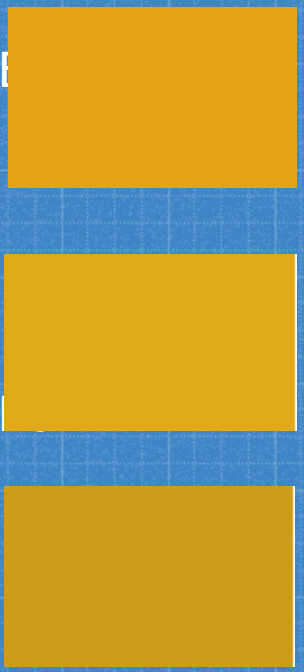
CREATION OF A CLOSE COLOR PAIR

- The 24-bit .bmp format actually stores pixels in B, G, R order
- Consider the following bitmap stream of 3 consecutive pixels:
- 15 A3 E4 16 AA E0 19 9D CC Embed:
- 1 0 1 1 0 1 0 0 1
  - 15 A3 E
  - 16 AA E
  - 19 9D CC



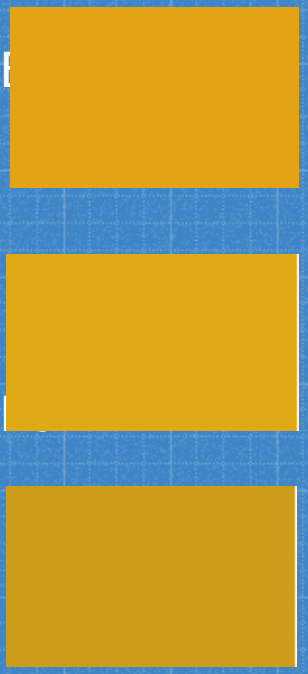
CREATION OF A CLOSE COLOR PAIR

- The 24-bit .bmp format actually stores pixels in B, G, R order
- Consider the following bitmap stream of 3 consecutive pixels:
- 15 A3 E4 16 AA E0 19 9D CC Embed:
- 1 0 1 1 0 1 0 0 1 Yields:
- 15 A2 E5 17 AA E1 18 9C CD
  - 15 A3 E
  - 16 AA E
  - 19 9D CC

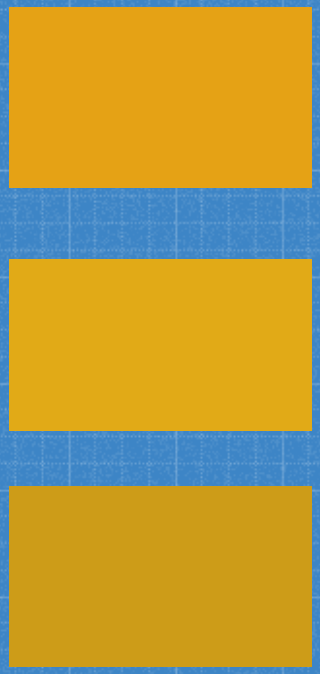


CREATION OF A CLOSE COLOR PAIR

- The 24-bit .bmp format actually stores pixels in B, G, R order
- Consider the following bitmap stream of 3 consecutive pixels:
- 15 A3 E4 16 AA E0 19 9D CC Embed:
- 1 0 1 1 0 1 0 0 1 Yields:
- 15 A2 E5 17 AA E1 18 9C CD
  - 15 A3 E
  - 16 AA E
  - 19 9D CC



- 15 A2 E5
- 17 AA E1
- 18 9C CD



CLOSE COLOR PAIR, DEFINED

- Two colors (R1, G1, B1) and (R2, G2, B2) are a *close pair* if
- **abs(R1-R2) ≤ 1 && abs(G1-G1) ≤ 1 && abs(B1-B2) ≤ 1**
- Or
- **$(R1-R2)^2 + (G1-G2)^2 + (B1-B2)^2 \leq 3$**
  - 15 A3
  - 15 A2
    - $0^2 + 1^2 + (-1)^2 = 2$
  - 16 AA
  - 17 AA
    - $(-1)^2 + 0^2 + (-1)^2 = 2$
  - 18 9C
    - $1^2 + 1^2 + (-1)^2 = 3$
  - 19 9D CC






The original colors are NOT close pairs.

3 Steganalysis Algorithm

ALGORITHM CREATORS

- Jiri Friedrich, Rui Du, Meng Long (SUNY Binghamton)
- “Steganalysis of LSB Encoding in Color Images”. IEEE, 2000

WHAT EXACTLY ARE WE TRYING TO DO HERE?

- The goal of this steganalysis example is to determine, at some confidence level, that a questioned image contains a hidden steganographic payload.

STEGANALYSIS ALGORITHM

- Let *U* = number of unique colors in an image
- Let *P* = number of close color pairs in the image
- Let *R* = the ratio of close color pairs to all color pairs
  - R = P/(U/2)
  - If R is “close” to 1, higher likelihood that steganography is present

(can adjust R threshold to change confidence interval)

VARIABLE CONFIDENCE

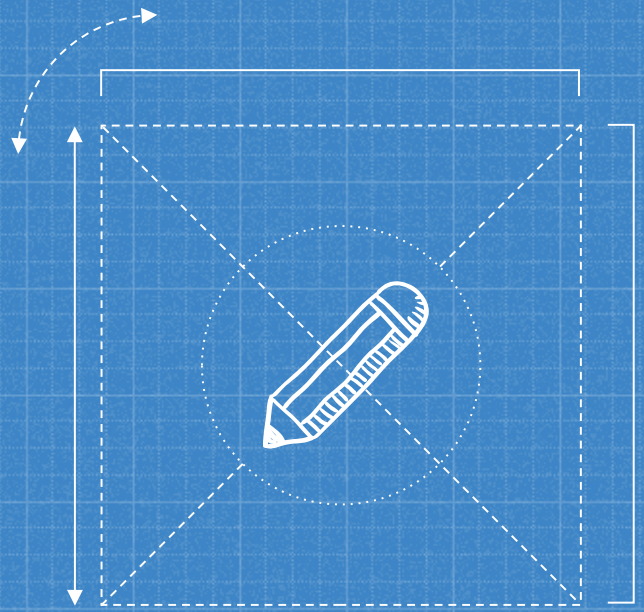
- The more unique colors in the questioned image, the lower the confidence of steganography detection via close pair ratio
- The bigger the size ratio of the payload file to the carrier file, the higher the confidence of detection

BUT THAT STILL ISN’T GOOD ENOUGH

- Depending on carrier size and encoding density, the ratio of close pairs to all pairs might still not be definitive
- Friedrich and company had a revelation

WHAT LSB STEGO DOES TO THE CARRIER

- Mentioned earlier: embedding a payload into LSBs essentially turn the stegchannel LSBs into random noise
  - Ratio R = (close pairs / all color pairs)
  - *Friedrich's Eureka Moment:*
  - If an image contains LSB stego, and you randomly flip the LSBs, the ratio of close pairs to all color pairs WON'T CHANGE MUCH FROM THAT OF THE ORIGINAL IMAGE
  - Ratio R' = (close pairs after random LSB change / all color pairs after LSB change
  - SO THE RATIO OF R TO R' IS YOUR DISTINGUISHER



BIG CONCEPT, YO

Calculate ratio R’ of close pairs to all pairs
Do it again after randomly flipping the LSBs
Then find the ratio of those 2 ratios

THE CLOSER THIS RATIO IS TO 1, THE HIGHER THE LIKELIHOOD THAT STEGANOGRAPHY IS PRESENT

MAN BEAR PIG

4

Carrier Selection and Stegchannel Creation

SELECTION OF A CARRIER IMAGE

- To optimize steganalysis, our ideal carrier file should be
  - Of moderate size compared to the payload
  - Consisting of a minimal amount of unique pixel colors
  - Does not contain an alpha channel

SELECTION OF A CARRIER IMAGE

- To optimize steganalysis, our ideal carrier file should be
  - Of moderate size compared to the payload
  - Consisting of a minimal amount of unique pixel colors
  - Does not contain an alpha channel

Stryper_noalpha_20colors.bmp
500x308 px
154000 pixels
151 KB

- This image has been posterized to use only 20 unique colors

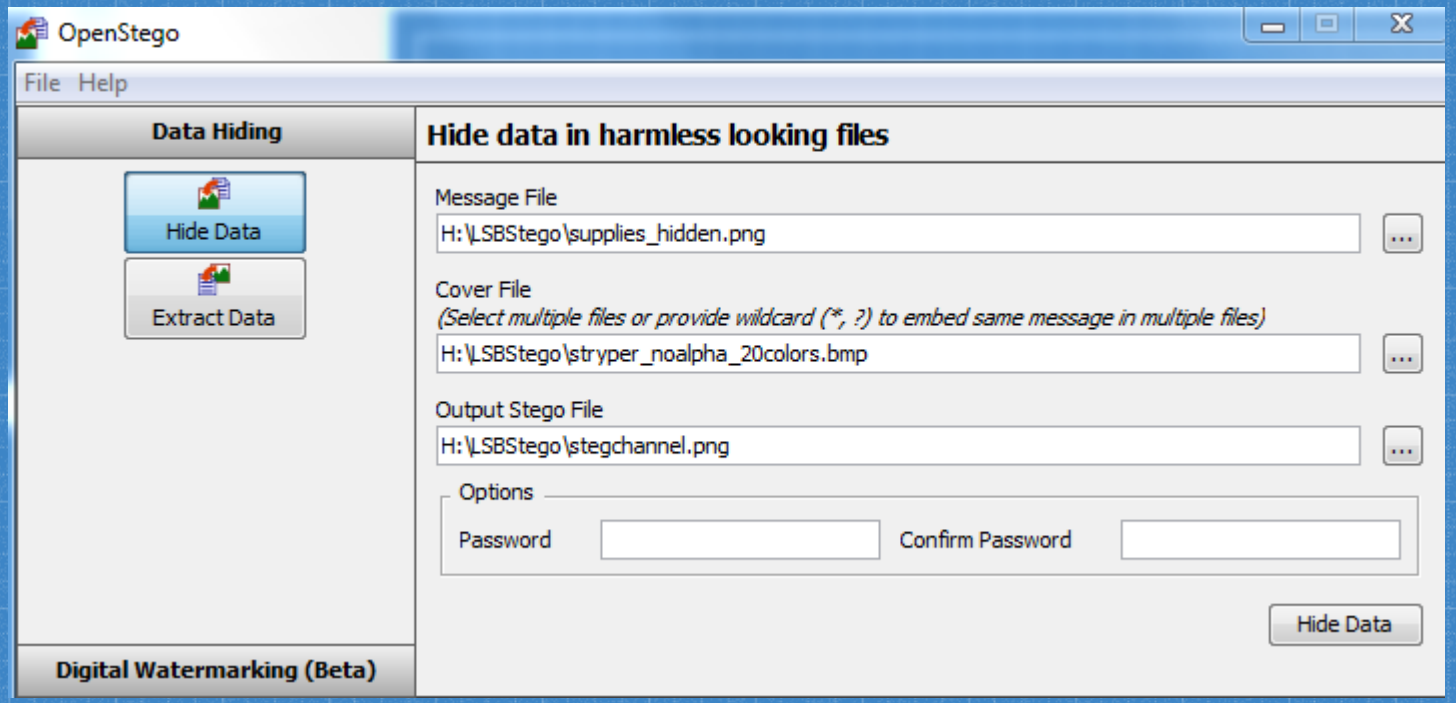
CHOOSE A PAYLOAD

- This could be ANY binary file
- In this case, we'll use a .png image

Supplies_hidden.png
225x225 px
99 KB
(65% of carrier image)

CREATE A STEGCHANNEL

- I used OpenStego to create my stegchannel
- Plenty of free/open source tools available
- Must be lossless



CARRIER VS STEGCHANNEL

- Carrier

151 KB BMP
20 unique colors

- Stegchannel

256 KB PNG
1280 unique colors

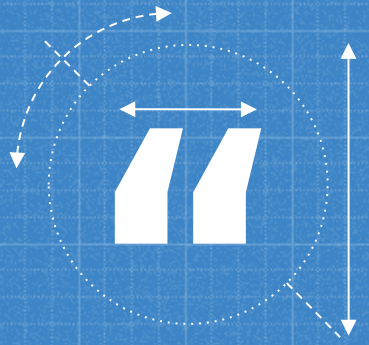
LET’S START THE CODE NOW

- The detection routine takes ~7mn to run. We’ll just let it do that in the background.

4
Detection Time!

STEGO DETECTION ALGORITHM

- Start with a suspected stegchannel image
- Calculate the number of unique colors in the image
- Serialize the image into a stream of bytes
- Group the byte stream into 3-byte RGB color channels
- Count all RGB values that differ by one, from the RGB values of any other pixel (close pairs)
- Calculate ratio of close pairs to total unique color values
- Randomly flip the LSBs and do it again
- If the ratio of these 2 ratios is “sufficiently” close to 1.0, higher confidence of stego

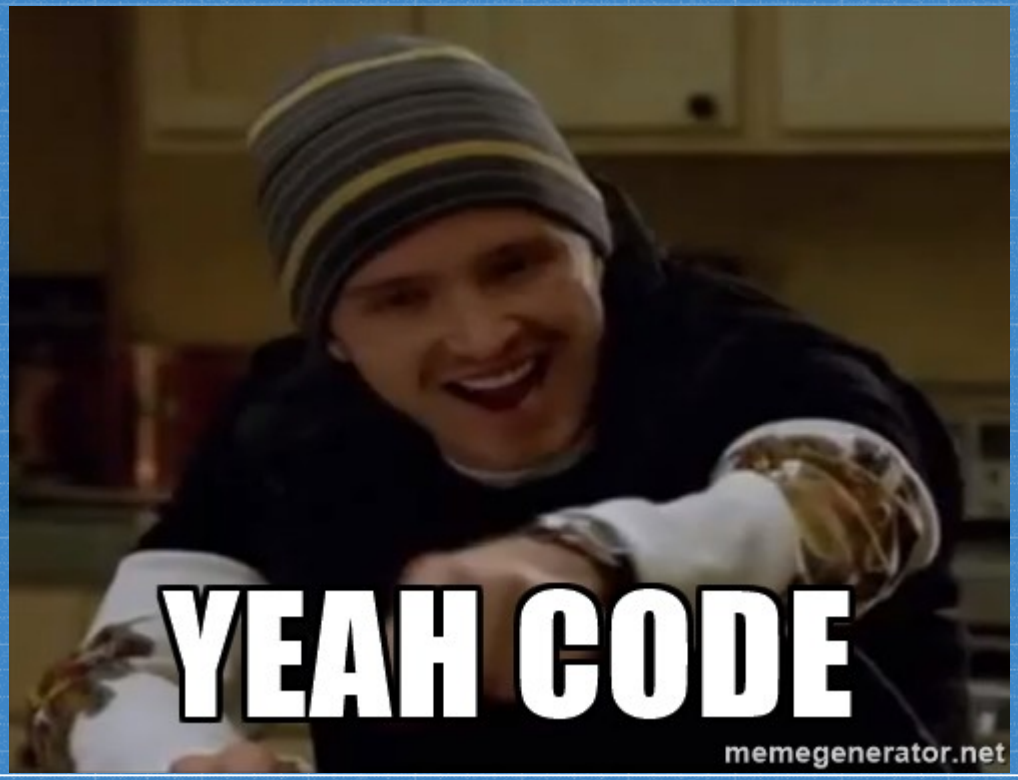


TIME FOR SOME POWERSHELL

FIRST: ACQUIRE IMAGE AND CONVERT TO A STREAM OF BYTES

- Use the –encoding switch of Get-Content to convert the file to a array of bytes

```
[Byte[]] $imageBytes =
get-content -encoding Byte
-path $file
```



NEXT: LOAD EACH COLOR CHANNEL INTO AN ARRAY

- Pass in the array of bytes
- R, G, B repeats every 3 bytes

```
for ($i = 0; $i -le ($imageBytes.Length - 3);
$i = $i +3)
{
    $redList.Add($imageBytes[$i])
    $greenList.Add($imageBytes[$i+1])
    $blueList.Add($imageBytes[$i+2])
}
```

RANDOMIZE ALL THE LSBs

- We now need to take the original byte array, and randomize all of its LSBs
- We then find the ratio of close color pairs, to all color pairs, for THIS image
- Powershell's Get-Random verb comes in handy here
- Binary XOR with a single bit will operate only on the LSB

```
foreach ($byte in $imageBytes)
{
    $random = Get-Random -Maximum 2
#returns 0 or 1

        if ($random -eq 1)
    # Binary XOR the LSB with 1 to
    # flip the LSB (otherwise leave    #
it the same)
    {
            $byte = $byte -bxor 1
    }
    $returnList.Add($byte)
}
```

NOW FIND ALL THE CLOSE PAIRS

- A “Close Pair” is any two pixels whose R, G, or B channels are the same, or off by one
- So: subtract the R, G, B values of 2 pixels, and square them to remove negative values
- If the sum of the squares of differences is less than or equal to 3, we have a Close Pair

- Do this for the original image, and the copy with randomized LSBs

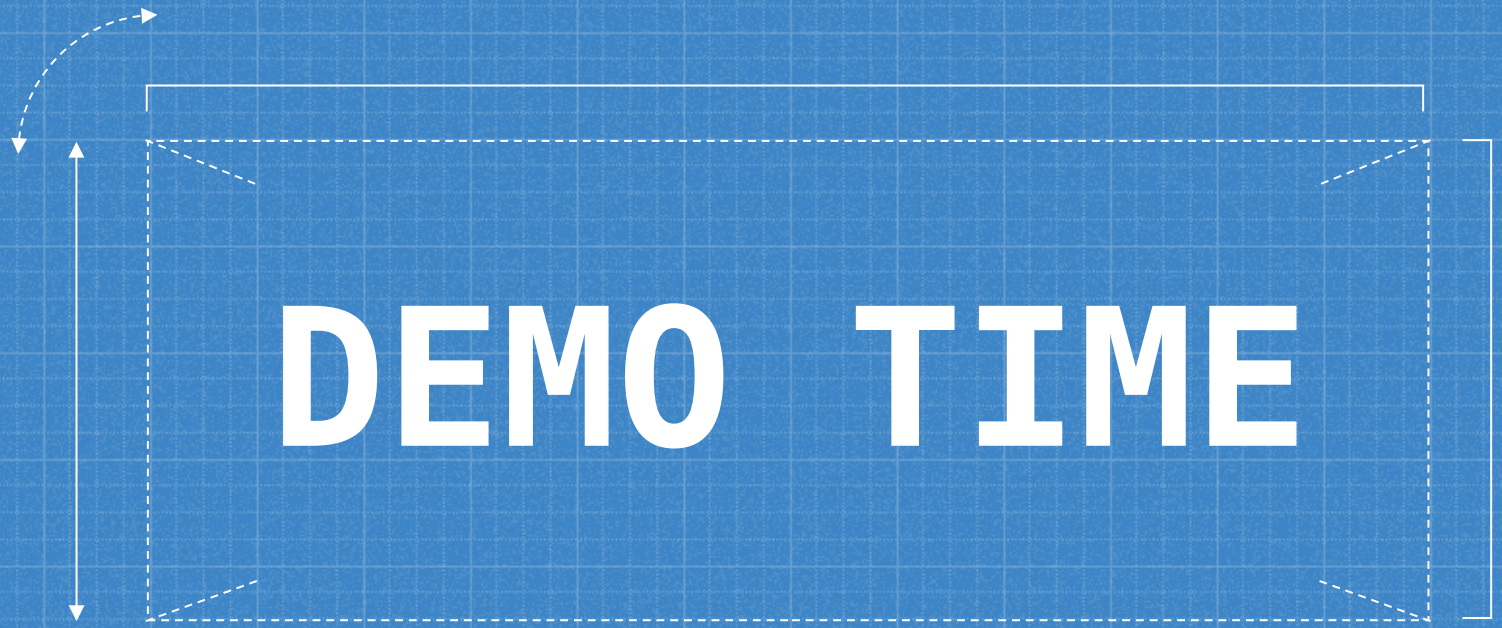
```
if
( ([math]::pow($deltaR,2) +
[math]::pow($deltaG,2) +
[math]::pow($deltaB,2) -le 3)
–and
([math]::pow($deltaR,2) +
[math]::pow($deltaG,2) +
[math]::pow($deltaB,2) -gt 0))

{
$closePairs++
}
```

MOMENT OF TRUTH

- Divide the close pair ratio for the original image with that of the randomized image
- If this ratio is “sufficiently close” to 1 (set by threshold), steganography is present

```
# If the ratio is within the range of
# +/- the difference between 1 and #
Threshold, likely that stego is
present
if ([math]::abs($masterRatio - 1) -le
([math]::abs($threshold - 1)) )
{
    write-host "Questioned image
$filePath contains LSB
steganography, at $threshold
confidence level."
}
```

BACK TO STRYPER


DEMO TIME

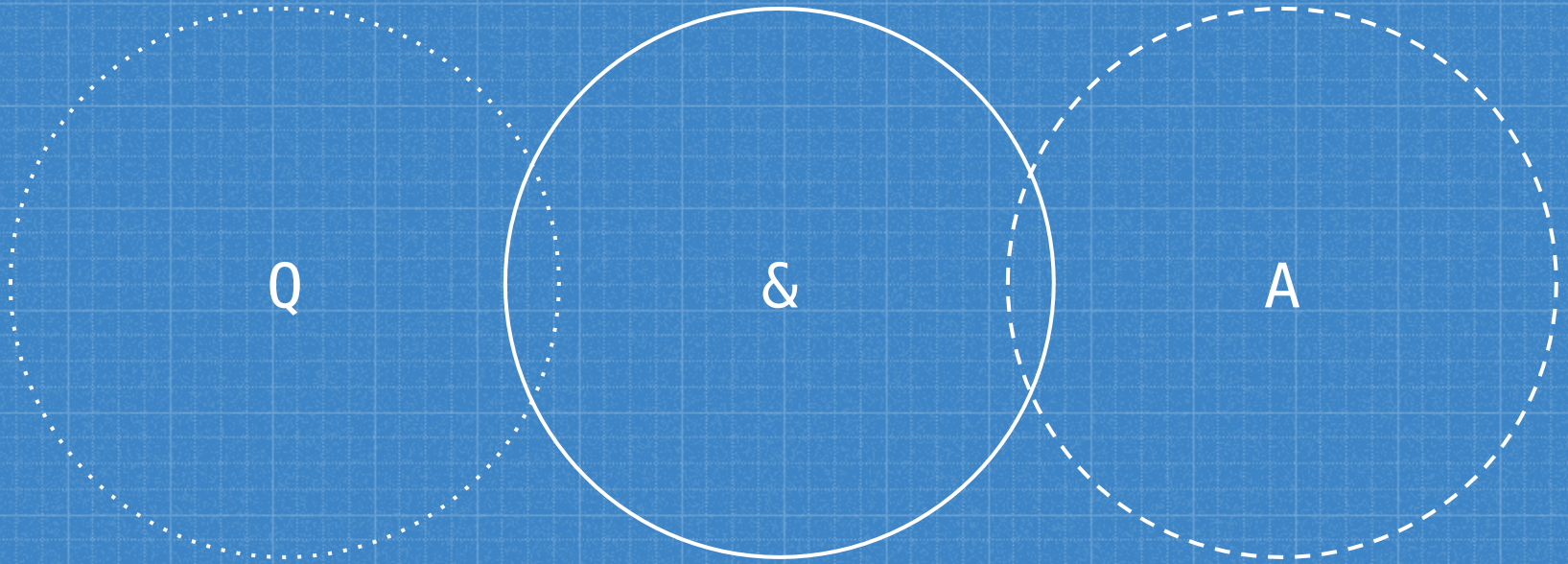
CREDITS

Special thanks to:

Fridrich et al, SUNY Binghamton. “Steganalysis of LSB Encoding in Color Images”. IEEE 2000.
http://ia.binghamton.edu/publication/FridrichPDF/icme00b.doc

Rubin, Brad, University of St. Thomas (who did this in Mathematica)

Want a copy of my code? joe.petroske@target.com


Q
&
A

THANK YOU

…and now it’s John Strand Time!